Elmar Schömer
Ann-Christin Wörl

8. Übungsblatt

Abgabe: Dienstag, der 19.12.2023, 14:00 Uhr

Aufgabe 1: Pyramide (5+5+5+5+5 Punkte)

Gegeben sei der untenstehende gerichtete Graph G. Wir wollen die Zahl $C(2n, 0)$ der verschiedenen Wege vom Punkt $(0, 0)$ zum Punkt $(2n, 0)$ für $n \in \mathbb{N}$ bestimmen. In der Grafik kann man zwei verschiedene Wege vom Punkt $(0, 0)$ zum Punkt (i, j) sehen.

1. Begründen Sie folgenden Zusammenhang für alle Punkte (i, j) mit $i + j$ gerade:

$$C(i,j) = \begin{cases} 1 & \text{für } i = j = 0 \\ 0 & \text{für } i = 0, j > 0 \\ C(i-1,1) & \text{für } i > 0, j = 0 \\ C(i-1,j-1) + C(i-1,j+1) & \text{für } i > 0, j > 0 \end{cases}$$

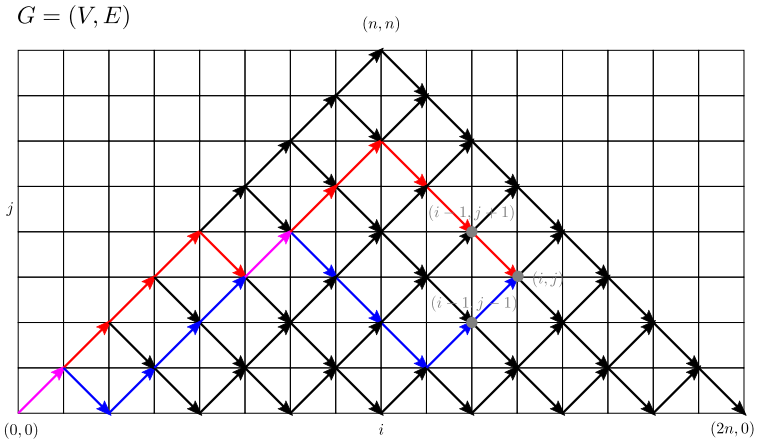
2. Schreiben Sie eine iterative (nicht-rekursive) Funktion zur Berechnung des Wertes $C(2n, 0)$. Wie wächst die Laufzeit Ihrer Funktion in Abhängigkeit des Parameters n?

3. Verifizieren Sie mit Hilfe Ihres Programmes, dass Folgendes gilt:

$$C(2n,0) = \frac{1}{n+1}\binom{2n}{n}$$

4. Schreiben Sie eine rekursive Funktion zur Berechnung des Wertes $C(2n, 0)$. Wie wächst die Laufzeit Ihrer Funktion in Abhängigkeit des Parameters n?

5. Nutzen Sie die Technik der Memoisierung zur Verbesserung der Laufzeit Ihrer rekursiven Funktion.

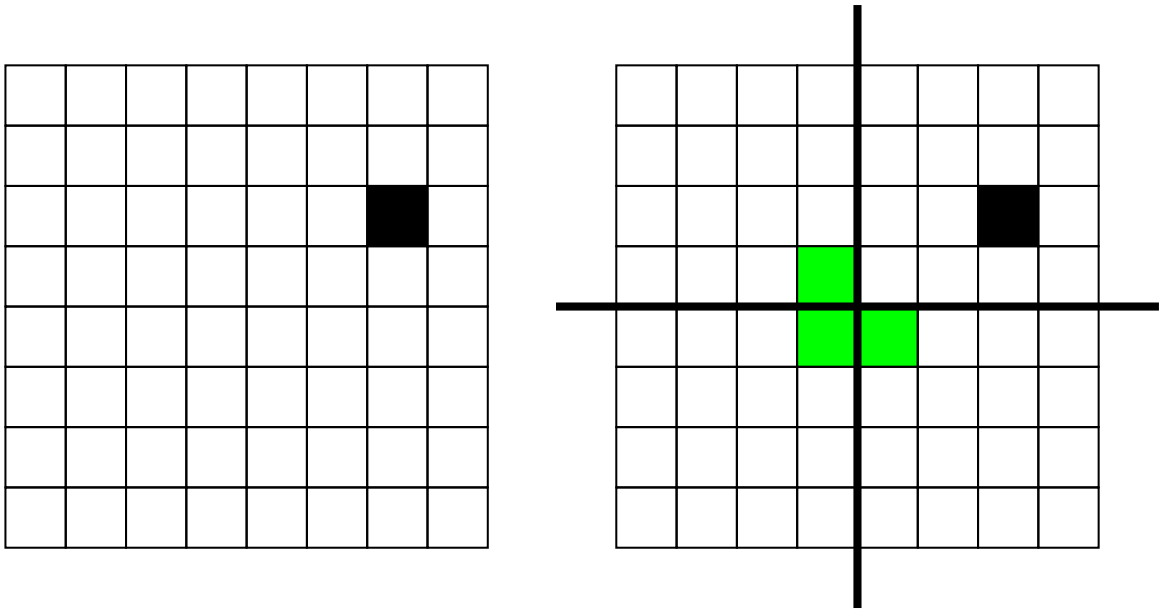


Aufgabe 2: Parkettierungsproblem (5+5+15 Punkte)

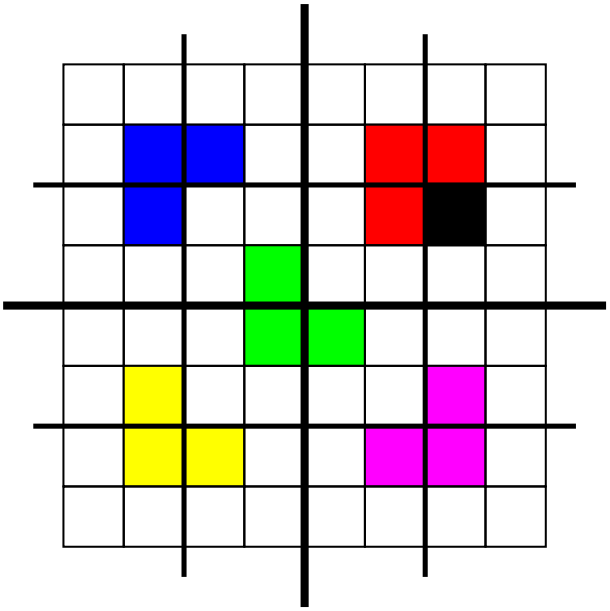
Wir wollen das folgende Parkettierungsproblem mit dem Teile-und-Herrsche Ansatz lösen. Gegeben ist ein zweidimensionales Feld **A** der Dimension $2^n \times 2^n$ für $n \in \mathbb{N}$. Ein Feldelement ist ausgespart. Die restlichen Feldelemente sollen mit L-förmigen Kacheln überdeckt werden, die je drei Feldelemente überdecken.

1. Eine notwendige Voraussetzung dafür, dass dies funktioniert, ist, dass die Zahl $(2^n)^2 - 1$ durch 3 teilbar ist. Zeigen Sie algebraisch, dass dies zutrifft.

2. Die Teile-und-Herrsche Strategie, um eine vollständige Parkettierung des Feldes zu berechnen, besteht darin, das Feld in vier Quadranten zu zerlegen und die erste Kachel so wie im Bild zu platzieren. Begründen Sie, dass sich aus dieser Vorgehensweise ableiten lässt, dass eine Parkettierung der gewünschten Art existiert.



3. Schreiben Sie eine rekursive Funktion `parkettiere(l,x,y)`, die das Feld im Bereich $[x, x+l-1] \times [y, y+l-1]$ parkettiert, wobei die Voraussetzung gelten soll, dass $l = 2^k$ ist und in dem betrachteten Bereich genau ein Feldelement als ausgespart betrachtet werden kann.



9	9	8	8	4	4	3	3
9	7	7	8	4	2	2	3
10	7	11	11	5	2		6
10	10	11	1	5	5	6	6
14	14	13	1	1	19	18	18
14	12	13	13	19	19	17	18
15	12	12	16	20	17	17	21
15	15	16	16	20	20	21	21

Aufgabe 3: Backtracking (10+10 Punkte)

1. Verändern Sie den Backtracking-Algorithmus zum Finden einer Lösung des n-Damen-Problems (siehe https://de.wikipedia.org/wiki/Damenproblem) so, dass alle möglichen zulässigen Stellungen berechnet werden. Nutzen Sie eine globale Variable, um diese Lösungen zu zählen.
2. Überprüfen Sie die berechnete Lösungsanzahl (ohne Berücksichtigung von Symmetrien) anhand folgender Tabelle und ergänzen Sie die Tabelle um die benötigten Laufzeiten.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$numSol$	1	0	0	2	10	4	40	92	352	724	2680	14200	73712	365596

```
1  def zulaessig(S):
2      k = len(S)
3      for i in range(k-1):
4          if S[i] == S[k-1]:
5              return(False)
6          if abs(S[k-1]-S[i]) == k-1-i:
7              return(False)
8      return(True)
9
10 def vervollstaendige(S):
11     if len(S) == n:
12         return(True)
13     for i in range(n):
14         S.append(i)
15         if zulaessig(S):
16             if vervollstaendige(S):
17                 return(True)
18         S.pop()
19
20     return(False)
21
22 n = 6
23 S = []
24 print(vervollstaendige(S))
25 print(S)
```

Aufgabe 4: Rucksackproblem II (5+10+5+10+10* Punkte)

Wir wollen noch einmal das Rucksackproblem (siehe https://de.wikipedia.org/wiki/Rucksackproblem) betrachten. Gegeben sind n Gegenstände. Der i-te Gegenstand besitzt den Wert w_i und das Gewicht g_i für $0 \leq i < n$. Die Tragelast des Rucksackes ist beschränkt durch das Gewicht G_{max}. Das Ziel ist es, den Rucksack mit einer möglichst wertvollen Auswahl von Gegenständen zu beladen, sodass das zulässige Gesamtgewicht nicht überschritten wird. Etwas formaler ausgedrückt ist eine Teilmenge $I \subseteq \{0, 1, \ldots, n-1\}$ gesucht, so dass

$$\sum_{i \in I} g_i \leq G_{max} \quad \text{und} \quad \sum_{i \in I} w_i \quad \text{maximal.}$$

Dieses Mal verfolgen wir einen etwas geschickteren Ansatz, als einfach alle möglichen Teilmengen I aufzuzählen. Dazu nehmen wir an, dass die Gewichte $g_i, G_{max} \in \mathbb{N}$ natürliche Zahlen sind. Wir gehen inkrementell vor, indem wir einen Gegenstand nach dem anderen in unsere Überlegungen einbeziehen. Angenommen wir wüssten bereits, wie man für die ersten $i-1$ Gegenstände (mit den Nummern $0, \ldots i-2$) und für verschiedene Tragelasten $\leq G_{max}$ eine optimale Auswahl trifft. Nun betrachten wir zusätzlich den Gegenstand mit der Nummer $(i-1)$ und fragen uns, ob dieser neue Gegenstand für eine optimale

Auswahl in Frage kommt. Wir definieren $R(i, G)$ als den größtmöglichen Wert, der sich mit einer Auswahl der ersten i Gegenstände (mit den Nummern $0, \ldots, i-1$) erzielen lässt, wenn die (verbliebene) Tragelast des Rucksackes G beträgt.

1. Begründen Sie die Korrektheit der folgenden rekursiven Definition:

$$R(i,G) = \begin{cases} 0 & \text{wenn } i = 0 \text{ oder } G = 0 \\ R(i-1,G) & \text{wenn } g_{i-1} > G \\ \max\{R(i-1,G), R(i-1,G-g_{i-1}) + w_{i-1}\} & \text{sonst} \end{cases}$$

2. Schreiben Sie eine rekursive Funktion zur Berechnung von $R(i, G)$, womit sich der gesuchte Wert $R(n, G_{max})$ berechnen lässt. Dabei sind wir nicht an der Auflistung der Gegenstände für eine optimale Auswahl interessiert, sondern nur an deren Gesamtwert.

3. Verwenden Sie Memoisation zur Beschleunigung Ihrer Berechnung. Bewerten Sie experimentell die Laufzeit Ihres neuen Programms im Vergleich zu der naiven Lösung.

4. Schreiben Sie eine Funktion zur Berechnung von $R[i][G]$, die vollständig auf Rekursion verzichtet.

5. **Zusatzaufgabe:** Ergänzen Sie Ihr Programm so, dass auch eine Auflistung der Gegenstände für eine optimale Auswahl möglich ist.